Программирование

Python

Структуры данных.
Черновик

Кафедра ИВТ и ПМ

2018

План

Прошлые темы

Пользовательские структуры данных

- Концепция записи
- Классы

Outline

Прошлые темы

- Что такое тип?
- Какие простые типы есть в Python?
- Какие составные типы есть в Python?

Outline

Логически связанные данные

Иногда требуется работать одновременно с несколькими отдельными но логически связанными переменами.

Например координаты точки на плоскости представляются двумя переменными.

Это вызывает большие неудобства если таких логически связаных переменных становится много. Например описания погоды (температура, облачность, осадки) в определённом городе, в определённый день.

Неудобства проявляются особенно сильно если нужно создавать ещё одну группу таких переменных для хранения данных, например для другого города.

Outline

Запись

Логически такие наборы данных можно описывать в виде набора полей.

Предположим требуется хранить данные о погодных условиях (температура, облачность, осадки) в определённом городе, в полдень на определённую дату.

```
Данные о погоде:
    Город
    ДеньГода
    Температура
    Облачность
    Осадки
```

Такое представление логически сгруппированных данных будем называть *записью*.

Концепция записей

ВЫПИСЬ ИЗЪ МЕТРИЧЕСКОЙ КНИГИ, ЧАСТЬ ПЕРВАЯ, О РОДИВШИХСЯ, ЗА ГОДЪ.

Счетъ родившихся: Мужеска пола	Женска пола	Месяцъ и день: рожд.	крещ.	Имена родившихся.	Званіе, имя, отчество и фамилія родителей, и какого вероисповеданія.	Званіе, имя, отчество и фамилія воспріемниковъ.	Кто совершалъ таинство крещенія.	Рукоприкладство свидетелей записи по желанію.
—	41.	Октября 25	31	Марія	Села Ново-Георгіевки приходской Священникъ Никодимъ Тихоновъ Синайскій и законная его жена Елизавета Михайлова, оба православные.	Села Михайловскаго Священникъ Павелъ Александровъ Ильинскій и села Ново-Никольскаго Священническая жена Екатерина Стефанова Никольская.	Священникъ села Михайловскаго Павелъ Ильинскій съ діакономъ Василіемъ Муратовымъ	

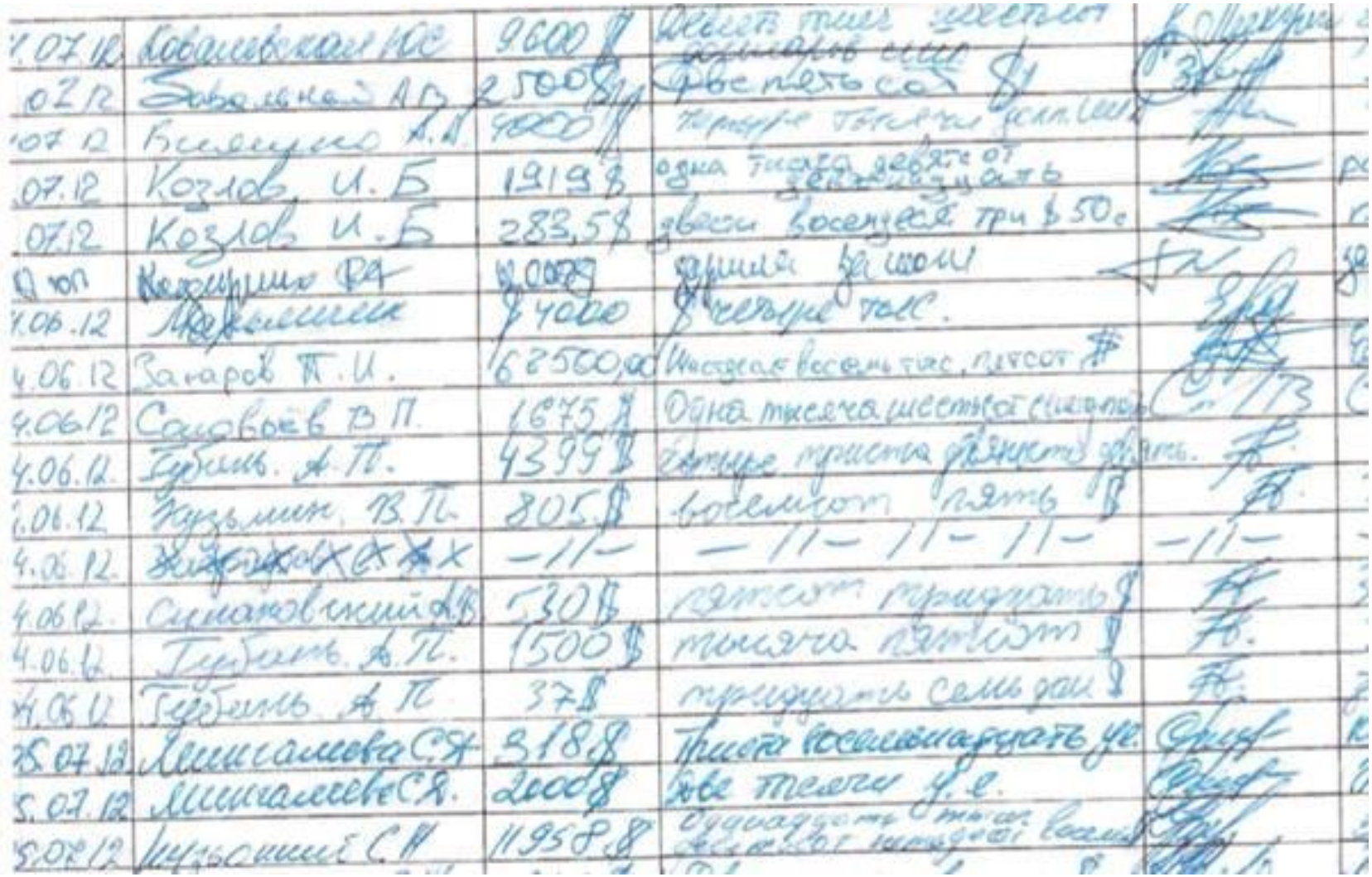
1917 г. 25 ...

Сія метрическая выпись во всемъ согласна съ копіей метрической книги за тысяча восемьсотъ девяностый четвертый годъ, ч. 1я, хранящейся при церкви села Ново-Георгіевки, что удостоверяемъ подписомъ и печатью.

1917 г. Іюля 25 дня
№ 179

Священникъ Николай Нечаевъ
Псаломщикъ ...

Концепция записей



Запись

Запись (record) — тип данных, набор значений различных типов.

Запись состоит из **полей**. Поле как было отмечено может быть представлено отдельным типом. В поле в том числе может быть другой записью.

Например поле *Дата* представляет собой отдельную запись состоящую из трёх полей.

```
Данные о погоде:
    Город
    Дата
        год
        месяц
        день
    Температура
    Облачность
    Осадки
```

Запись

В языках программирования записи могут быть представлены разными способами.

В Pyhton запись можно представить с помощью одного из составных типов данных: *кортежа, списка или словаря*.

Другой способ представления записи - тип данных **класс (class)**.

Хотя понятия класса включает в себя ещё и методы (операции производимые с данными), его можно использовать только для хранения данных.

Outline

Классы и объекты

Класс — составной **тип данных**, который может быть описан программистом.

Объект - экземпляр класса; *переменная* типа класс.

Классы и объекты

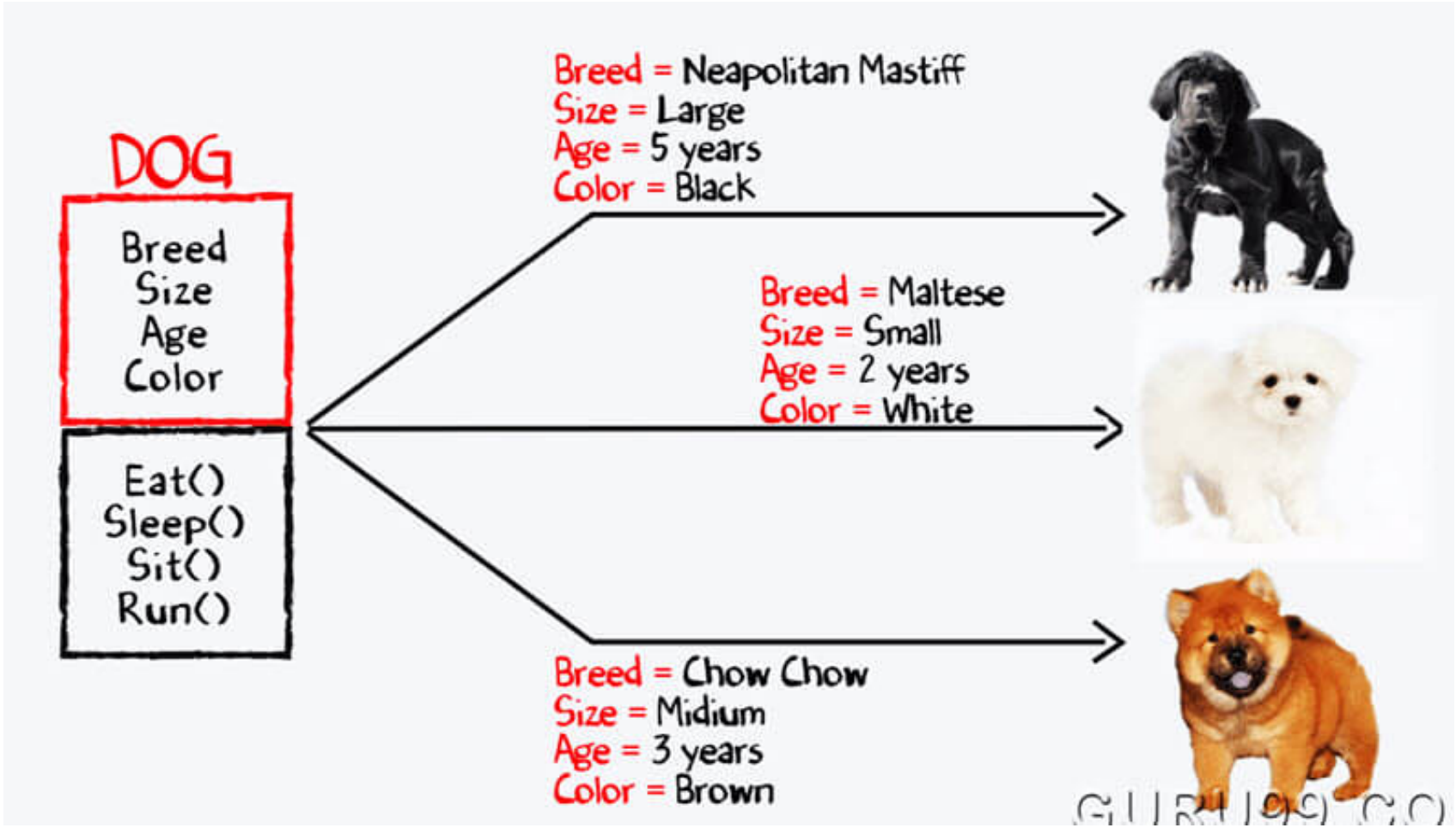
Классы могут включать в себя переменные других типов - **поля** (свойства).

Кроме того, класс может включать в себя функции - **методы**.

Класс как тип данных представляет собой только набор полей которым не заданы конкретные значений и набор методов.

Во время создания объекта (объявления переменной типа класс) этим полям задаются конкретные значения.

Классы и объекты



Класс как контейнер данных

Представим запись *Дата* в виде класса (создадим новый тип данных).

```
class Date:
    day = 1
    month = 1
    year = 1
```

В Python нельзя объявить поле, только определить: привести идентификатор и значение.

Когда будет создана переменная типа *Date* её поля уже будет содержать указанные при определение класса значения.

Если нужно отличать переменную с заполненными значениями полей, от аналогичной незаполненными то в качестве начальных значений используют None.

Класс как контейнер данных

Создание переменной описанного типа Date.

Переменная типа класс называется **объектом** или **экземпляром класса**.

```
my_birthday = Date()
othder_date = Date()
d1 = Date()


type( d1 )  # __main__.Date
```

Каждый описанный класс представляет собой отдельный тип данных.

С технической точки зрения имя класса в python включает ещё и имя пространства имён. В приведённом примере пространство имён называется __main__ [1]

[1] Имена переменных начинающиеся и заканчивающиеся символа подчёркивания играют роль служебных ("для внутреннего пользования"), и непосредственное использование таких имён не рекомендуется

Класс как контейнер данных

Чтобы получить доступ к полям используется **селектор** - оператор "."(точка).

```
my_birthday = Date()

my_birthday.day = 31
my_birthday.month = 1
my_birthday.year = 1956
```

Класс как контейнер данных

В Python любой тип являются классом, в том числе простые типы.

Однако часто с точки зрения программиста работа с переменными встроенных типов (int, float, tuple и т.д.) выглядит точно так же как и с обычными переменными (в других языках).

Класс как контейнер данных

```
my_birthday = Date()

my_birthday.day = 31
my_birthday.month = 1
my_birthday.year = 1956

# отдельные экземпляры класса независимы
d = Date()
d.year = 1984

print( my_birthday.year )  # 1956
print( d.year )  # 1984
```

Таким образом для логически связанных наборов данных следует создавать либо классы либо организовывать их с помощью встроенных типов: списков, кортежей, словарей.

Такой подход позволяет логически организовать данные, уменьшить количество отдельных переменных.

Ссылки и литература

Ссылка на слайды

github.com/VetrovSV/Programming